Journal of King Saud University – Computer and Information Sciences 35 (2023) 101606

HOSTED BY

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

Contextual Embeddings based on Fine-tuned Urdu-BERT for Urdu threatening content and target identification

Muhammad Shahid Iqbal Malik [a,b,*], Uswa Cheema [b], Dmitry I. Ignatov [a]

[a] Department of Computer Science, National Research University Higher School of Economics, 11 Pokrovskiy Boulevard, Moscow 109028, Russian Federation
[b] Department of Computer Science, Capital University of Science and Technology, Kahuta Road Sihala, Islamabad, Pakistan

ARTICLE INFO

Article history:
Received 1 March 2023
Revised 30 May 2023
Accepted 30 May 2023
Available online 5 June 2023

Keywords:
Threatening content
Target identification
Fine-tuned BERT
Urdu
Twitter
Text representation

ABSTRACT

Identification of threatening text on social media platforms is a challenging task. Contrary to the high-resource languages, the Urdu language has very limited such approaches and the benchmark approach has an issue of inappropriate data annotation. Therefore, we present robust threatening content and target identification as a hierarchical classification model for Urdu tweets. This study investigates the potential of the Urdu-BERT (Bidirectional Encoder Representations from Transformer) language model to learn universal contextualized representations aiming to showcase its usefulness for binary classification tasks of threatening content and target identification. We propose to exploit a pre-trained Urdu-BERT as a transfer learning model after fine-tuning its parameters on a newly designed Urdu corpus from the Twitter platform. The proposed dataset contains 2,400 tweets manually annotated as threatening or non-threatening at the first level and threatening tweets are further categorized into individual or group at the second level. The performance of fine-tuned Urdu-BERT is compared with the benchmark study and other feature models. Experimental results show that the fine-tuned Urdu-BERT model achieves state-of-the-art performance by obtaining 87.5% accuracy and 87.8% F1-score for threatening content identification and 82.5% accuracy and 83.2% F1-score for target identification task. Furthermore, the proposed model outperforms the benchmark study.

1. Introduction

The development of communication technology has enabled social media platforms to play a key role in our daily lives. The number of social media users is growing exponentially, doubled from 2.73 billion to approximately 4.26 billion from 2017 to 2021 (Statista). Recent statistics revealed that more than 486 million users are active on Twitter and more than 1.4 trillion tweets are posted annually. Twitter is a commonly used social network, that facilitates tweets up to 280 characters at maximum. It provides ease and freedom of speech, however, some users exploited Twitter to engage themselves in several crimes such as phishing and malware dispersion (Schmidt and Wiegand, 2017; Wang et al., 2015). Furthermore, these crimes raised some serious concerns and challenging issues like violence based on gender, threats to individuals or groups, and physical violence (Del Vigna12 et al., 2017).

Unrestricted access to social media (like Facebook, Twitter, etc) has aroused serious concerns in society. In the last decade, abusive/offensive content, the use of threatening language, and violence incitation have become predominant in the mainstream of social media. To handle these issues, many studies are carried out and proposed significant solutions like cyberbullying detection using random forest with psychological features (Balakrishnan et al., 2020), and a convolutional neural network (CNN) based model for cyber threats detection (Behzadan et al., 2018). Furthermore, research work on the detection of propaganda (Malik et al., 2023) and abusive speech (Nelatoori and Kommanti, 2022) is carried out mainly for high-resource languages like English, etc. Likewise, some latest approaches addressed hate speech detection in English (Mazari et al., 2023; Kamal et al., 2023; Saeed et al., 2023). In contrast, few approaches focused on low-resource

* Corresponding author at: Department of Computer Science, National Research University Higher School of Economics, 11 Pokrovskiy Boulevard, Moscow 109028, Russian Federation.
E-mail address: mumalik@hse.ru (M.S.I. Malik).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2023.101606
1319-1578/

languages like hate speech, and offensive language detection in Urdu (Hussain et al., 2022; Beyhan et al., 2022).

As far as threatening content detection is concerned, it was explored in English (Ashraf et al., 2020; Jiang et al., 2023; Mazari et al., 2023) as well as in other languages like Arabic, Indonesian, Bengali, etc (Mubarak et al., 2022; Febriana and Budiarto, 2019; Wadud et al., 2022). These studies exploited linguistic, syntactic, lexical, and semantic characteristics to derive detection models. We found very little effort in the literature for threatening text detection and target identification in the Urdu language. According to our knowledge, Amjad et al. presented the first detection method for threatening content and then target identification as a hierarchical classification (Amjad et al., 2021b). We did not find any other research work on this subject however few tried to handle only the threatening content identification but not target identification in Urdu (Humayoun, 2022; Mehmood et al., 2022). To fulfill this gap, two objectives are devised in this study:

1. To design an automated identification system for Twitter data to classify Urdu tweets as threatening or non-threatening accurately.
2. For threatening tweets; design an effective framework to identify the type of target/community (individual or group) being victimized.

We found an issue of inappropriate annotation (see section 3.1) in the dataset used by the study (Amjad et al., 2021b), that's why their method achieved an accuracy of 75% at maximum with the balanced dataset. To fill out this research gap and the limitations of prior studies, our work contributes in two directions: 1) proposes a new dataset manually annotated by three annotators, and 2) utilizes the fine-tuning of the BERT model for threatening content and target identification for the Urdu tweets. The contributions of the current study are given below:

- This study devises a significant threatening and target identification framework using contextual semantic embeddings to handle the ambiguity and complexity issues of the Urdu language.
- An Urdu corpus for threatening content and target identification detection is built by extracting tweets from Pakistani user accounts and annotating them manually by experts.
- One of the major contributions is the fine-tuning of Urdu-BERT for two distinct tasks; threatening content and threat target identification in the Urdu language.
- Extensive experiments are performed to optimize the hyperparameters for Urdu-BERT to improve classification performance as well as to handle catastrophic forgetting and overfitting issues.
- Several comparative experiments are conducted to showcase that the fine-tuning of Urdu-BERT demonstrates high performance in comparison with the benchmark study.

The remaining parts of the study are organized as follows: Section 2 presents the related work carried out in several languages and their comparison in tabular form. Section 3 describes the proposed method in detail and the dataset collection and annotation mechanism. Section 4 provides the results and analysis and discusses different aspects of the proposed framework. The last section describes the conclusion and future directions.

2. Related work

This section describes the literature work done for abusive and threatening content detection in Urdu and other languages. It is quite challenging to separate threatening text from negative/abusive content. Amjad et al. introduced the first method in this domain by developing an Urdu corpus for threatening content and target identification tasks (Amjad et al., 2021b). They utilized word and char n-grams and FastText embedding features with some machine learning (ML) and two Deep Learning (DL) methods. Multilayer Perceptron (MLP) model with word n-gram obtained the best results for threatening content and Support Vector Machine (SVM) with FastText obtained the best performance for target identification. As our aim is also threatening content and target identification from tweet text, therefore we have chosen their study as the benchmark for comparing the effectiveness of our model.

Later, a study developed a detection framework for threat records in Urdu (Kalraa et al., 2021) by using DL methods but their dataset is highly imbalanced. Likewise, Das et al. (Das et al., 2021) built an abusive and threatening text detection model in the Urdu language by exploring BERT, XGBoost, and Light Gradient Boosting Machine (GBM) models. Their model showed an F1-score of 0.88% and 0.54% for abusive and threatening language detection. Then another abusive content and threatening text detection model is built (Humayoun, 2022) by using word n-gram and word2vec feature models but their model did not achieve a significant F1-score (49.31%). Recently Mehmood et al. (Mehmood et al., 2022) designed a detection model by comparing several ML and DL models. Their results revealed that the stacking-based ensemble approach obtained an F1-score of 73.99%.

In the Urdu language, some studies focused on the detection of abusive/offensive content from social media platforms such as Hussain et al. designed a detection model for offensive language in Urdu for the Facebook platform (Hussain et al., 2022). Their model achieved 88.27% accuracy with word2vec and a voting-based ensemble model. Another study (Amjad et al., 2021a) presented a detection model for abusive text detection for the Twitter network using char and word n-grams and FastText embedding models. Their experiments revealed that char tri-gram with SVM obtained 82.68% F1-score. Another study (Akhter et al., 2021) proposed a detection approach for abusive language in Urdu and Roman Urdu. They used four ML and four DL models with Bag-of-Words (BOW) features. The best results revealed an accuracy of 96% with the CNN model. Recently, Saeed et al. (Saeed et al., 2023) built a detection model to identify offensive text and its severity in the Urdu language. They utilized BERT, char n-gram, word n-gram, and word embeddings and their model demonstrated 86% F1-score.

In the English language, many studies addressed the issue of unwanted/abusive content identification from social media, e.g., a study (Razavi et al., 2010) developed a flame detection model by utilizing a three-level classification strategy to complete the task with naïve Bayes classifier. Later, Rani and Ojha exploited the word n-gram model to derive an offensive text detection framework in English (Rani and Ojha, 2019). Authors used up to word 8-grams to design three sub-systems and achieved 79.76%, 87.91%, and 44.37% accuracy. According to our knowledge, threatening language and target identification in the English language was explored for the first time by (Ashraf et al., 2020). They proposed a framework using BOW, Glove, and FastText feature models with CNN, LSTM, and Bi-LSTM methods. Their model obtained an 85% F1-score. Later, an hate speech detection model for the English language is proposed by (Saleh et al., 2021). The authors utilized BERT with the LSTM model to design the detection system. Their results revealed that the model demonstrated more than 90% F1-score on four datasets. Recently, Hajibabaee et al. proposed a detection framework for offensive language in English (Hajibabaee et al., 2022). They explored three embedding methods and eight

classifiers, and their model demonstrated a benchmark performance with a 95% F1-score.

Researchers also proposed solutions for hate/abusive speech detection in other languages like Turkish, Bengali, etc. A hate speech detection framework was proposed by (Beyhan et al., 2022) for the Turkish language. Their study prepared a hate speech corpus and developed a BERTTurk architecture. Results showed the highest accuracy of 77%. Similarly, hate speech detection in the Indonesian language is proposed by (Febriana and Budiarto, 2019). They released a new dataset and utilized latent dirichlet allocation and sentiment analysis to generate polarity scores for each tweet. Then, Sigurbergsson and Derczynski proposed a hate speech and offensive text detection model in the Danish language (Sigurbergsson and Derczynski, 2019), by exploring linguistic, pre-trained embeddings, word uni-gram, and sentiment features. Their model demonstrated significant performance. Likewise, another study (Wadud et al., 2022) built an offensive content detection system for the Bengali language using word embedding vectors with the LSTM-BOOST model. Their model obtained a 92.61% F1-score on the Bengali dataset and outperformed the baseline. For the Arabic language, a study (Mubarak et al., 2022) investigated the impact of emojis in the identification of hate speech and offensive content from tweets. The authors used emojis, linguistic words, and AraBERT models. Their framework achieved 92.99% accuracy. In the end, the summary of literature work is presented in Table 1.

The Urdu language has very limited contributions while addressing the tasks of threatening content and target identification. Only Amjad et al. addressed the two tasks (threatening content and target detection) for the Urdu language. We found deficiencies in the literature in the following directions:

- **Corpus in Urdu Language:** One dataset is available but it suffers from the issue of inappropriate annotation.
- **Effective feature engineering methods**: The majority of studies applied syntactic and lexical features such as word and char n-grams etc.
- **Comparison between classifiers:** In the majority, prior studies missed the comparison between machine learning and deep learning models to determine the best classifier.

3. Proposed architecture

This section presents the details of the sub-steps adopted to design the proposed framework for the identification of threatening language and corresponding target identification from the tweets. The sequence of steps is illustrated in Fig. 1.

3.1. Designing the corpus and labeling

This study uses tweets to evaluate the effectiveness of the proposed identification framework. Although a prior study (Amjad et al., 2021b) designed and annotated a dataset to achieve the same objectives, we found issues in the labeling mechanism of their dataset for threatening and non-threatening classes. The majority of their threatening labels are actually abusive but they considered them threatening. We added a few examples in Table 2, here we can analyze that all tweets are actually abusive but they labeled them as threatening.

Due to these inconsistencies, we designed a new dataset to meet our requirements of threatening content and target identification tasks. The data was collected from Twitter using Twitter API. In the first step, we developed a lexicon of seed words to spot the relevant tweets. This lexicon was created manually and consists of 250 keywords. Few keywords contain one word whereas the remaining contain two, three, or four words. The examples of a few keywords are presented in Table 3.

Using the lexicon, we collected only those tweets which contain at least one of the keywords of the lexicon. The collection process is illustrated in the left part of Fig. 2. The time period of 24 months ranging from August 2020 to August 2022 is selected for tweets collection. The reason why we chose this time period is that the political situation was very unstable in Pakistan. The process of

Table 1
Summary of prior studies related to threatening, hate speech, and abusive text identification.

Language [ref]	Tasks			Feature Extraction	Supervised Methods
	Abusive	Threat	Target		
English (Razavi et al., 2010)	√			Phrases and keywords	NB
English (Rani and Ojha, 2019)	√			Uni-gram	SVM, Bi-LSTM, CNN
Indonesian (Febriana and Budiarto, 2019)	√			Latent Dirichlet Allocation	–
Danish (Sigurbergsson and Derczynski, 2019)	√		√	Word n-grams, BOW, Linguistic, pre-trained embeddings	Fast-LTSM, Bi-LSTM, LR
English (Ashraf et al., 2020)		√	√	BOW, Glove, FastText	CNN, LSTM, Bi-LSTM
English (Hajibabaee et al., 2022)	√			TF-IDF, word2vec, FastText	MLP, SVM, AdaBoost, RF, LR, DT, NB
Bengali (Wadud et al., 2022)	√			Word embeddings	LSTM-BOOST, PCA, AdaBoost
Turkish (Beyhan et al., 2022)	√		√	BERT	BERT Classifier
Urdu, Roman Urdu (Akhter et al., 2021)	√			BOW	NB, DT, KNN, SVM, CNN, LSTM, Bi-LSTM, C-LSTM
English (Saleh et al., 2021)	√			BERT	BERT-LSTM
Urdu (Kalraa et al., 2021)		√		Transformer model	DL
Urdu (Amjad et al., 2021b)		√	√	Char and word n-grams, FastText model	MLP, AdaBoost, RF, LR, SVM, CNN and LSTM
Urdu (Das et al., 2021)	√	√		BERT model	XGBoost, LightGBM, mBERT, dehateber-mono-arabic
Urdu (Amjad et al., 2021a)	√			Word and char n-grams, FastText Embeddings	(RF, LR, SVM, AdaBoost, CNN, LSTM
Urdu (Mehmood et al., 2022)		√		TF-IDF, BOW	Stacking ensemble, RF, SVM, CNN, LSTM, GRU
Urdu (Hussain et al., 2022)	√			TF-IDF, BOW, Word n-grams, word2vec	Stacking based ensemble, LR, RF, SGD, SVM
Urdu (Humayoun, 2022)		√		BOW with TF-IDF, word2vec	SVM, AdaBoost, CNN
Arabic (Mubarak et al., 2022)	√			Emoji, Linguistic terms, BERT	SVM, BERT, AraBERT
Urdu (Saeed et al., 2023)	√			BERT, char and word n-grams, word embedding	SVM, LR, CNN, LSTM

BOW: bag of words; CNN: Convolutional Neural Network; LSTM: Long Short Term Memory; GRU: Gated Recurrent Unit; MLP: Multi-layer perceptron; RF: Random forest; SVM: Support vector machine; LR: Logistic Regression; SGD: Stochastic Gradient Descent; DT: Decision Tree; NB: Naïve Bayes.

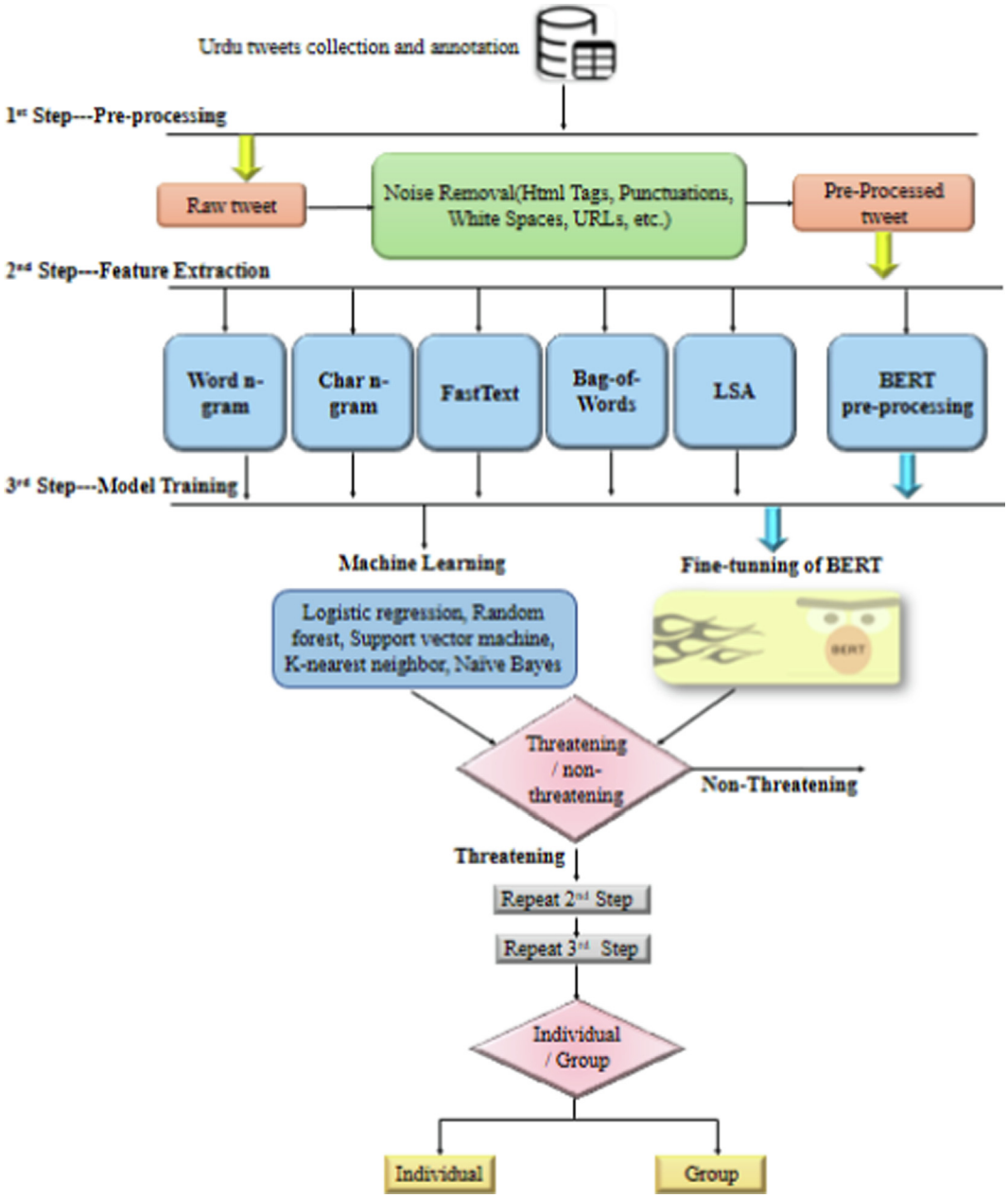

Fig. 1. Pipeline of the proposed architecture.

collection of tweets was stopped at 20,000 samples. After that, cleaning of data was performed by identifying and discarding tweets having empty text, duplicate content, and tweets containing other language words. After that, another cleaning step is applied to the data corpus; we know that the Urdu language takes words from other languages such as Persian and Arabic. Therefore,

Table 2
Examples of threatening tweets by the prior study (Amjad et al., 2021b).

Urdu	English Translation
بکواس مت کرو	Shut up
یہ عورت کے نام پر بدنامی کا دھبہ ہے	It is a stain on the name of a woman
ہم آخری سانس تک پاک فوج کے ساتھ ہیں	We are with Pakistan Army till the last breath
گانڈو یہ ویڈیو کسی فلپینی بچی کی ہے۔ تم لوگ ایسا کرو اپنے کپڑے اتھار دو	Bottom, this video is of a Filipino girl. You guys do that, take off your clothes
ان کھوتی کے بچوں کی اوقات بہی ہے سالوں نے میری فیسبک کی ایڈی ایک ماہ کے لیے بلاک کردے ٹویٹر کے تین اکاونٹس بھی	This is the reality of these kids of ass. They have blocked my Facebook account for a month and also three Twitter accounts.
تجے بڑی فکر ہے بہن دی ہونی ہے پاکستان کو	Why are you so worried, did you give your sister to Pakistan

Table 3
Examples of Seed words.

Urdu	Translation	Urdu	Translation	Urdu	Translation
ٹکڑے	Pieces	افسوسناک موت	Sad death	خون کے آنسو	Tears of blood
ہلاک	Killed	ٹانگیں توڑ	Break the legs	عبرت کا نشان	Sign of lesson
لاشیں	Corpses	چھورا گھوپنا	Stabbing	خون پی جاؤں	Drink blood
دھمکی	Threat	چیر دوں	Tear	جان سے مار دو	Kill with soul
جنازہ	Funeral	الٹا لٹکا	Hang upside	سر تن سے جدا	Head separated from body

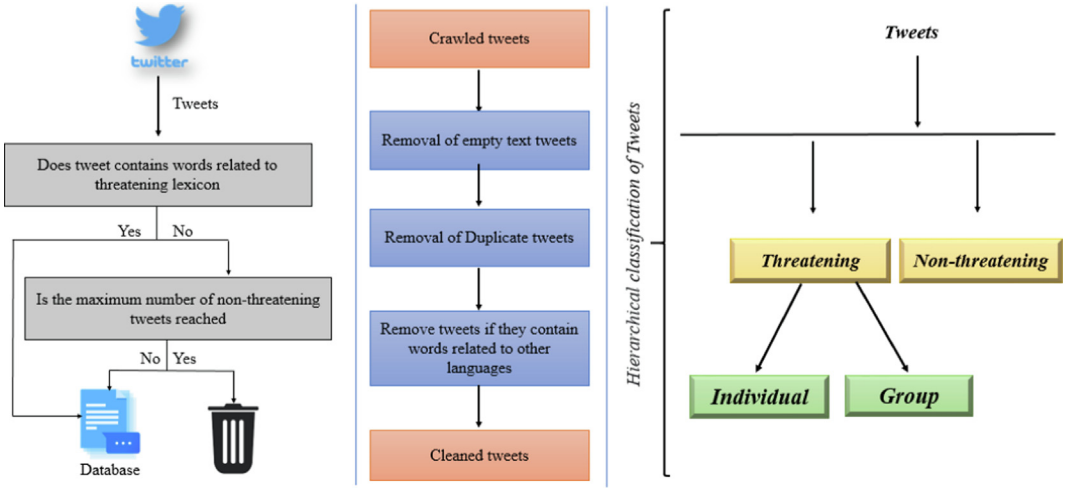

Fig. 2. Tweets collection, cleaning process, and hierarchical classification of tweets.

those tweets are removed which contain Persian/Arabic words. The cleaning process is illustrated in the middle part of Fig. 2. After cleaning, dataset annotation was performed. We took the services of three annotators from Pakistan who satisfied the following criteria: 1) Native Urdu speaker, and 2) Have at least MS level education and prior experience of annotating Urdu data. This research aims to perform two tasks: first; the classification of tweets into threatening or non-threatening and second; threatening tweets are further categorized into individual or group. This hierarchical classification is illustrated in the right part of the Fig. 2.

To address two tasks, appropriate annotation guidelines are designed and provided to the annotators to help them in labeling the tweet data. At the first level, annotators need to categorize the tweets into threatening or non-threatening by analyzing the tweet text. There could be two scenarios, e.g. some people try to threaten a person by using abusive words/phrases as shown in the first example given in Table 4. Sometimes people used normal words to threaten someone as illustrated in the second example. After the first level of categorization, annotators need to analyze the body of threatening tweets to distinguish whether an **individual** is targeted or a **Group** (more than one).

In literature, hidden biases are reported in the data sampling/-collection process like the study (Shang, 2017) disclosed some biases. Our collection methodology and data are free of such biases. We used Fleiss' kappa inter-annotator agreement (Fleiss, 1971) to compute the agreement between annotators. As there are binary labels for both types of tasks, the overall kappa coefficient is 80%. It is a common issue with every dataset that instances of the positive class are less than the negative class. The same is the issue here, non-threatening tweets are more than threatening tweets but we are interested in a balanced dataset. Hence our dataset consists of 1200 threatening and 1200 non-threatening instances. This completes the process of corpus construction.

3.2. Bert model

Among the language models, BERT has proved its significance for a wide range of natural language processing and text mining tasks (Malik et al., 2023). Specifically, the benefits include automated contextual feature generation, fewer data requirements, faster development, and improved performance. It is a pre-trained model developed by the researcher at Google Lab (Devlin et al., 2018). BERT can be used in two ways: First, it can be utilized as a feature extractor model by recognizing the text and the context of the input data, and Second, fine-tuning the pre-trained model to achieve the best performance by absorbing and identifying the

Table 4
Examples (guidelines) for annotating the tweets.

S#	Urdu	Translation	Level 1	Level 2
01	لعنتی کتے عوام کا حال دیکھو لوڈشیڈنگ کو روکو خدا کی قسم ہم تم کو زندہ جلا دیں گے	Cursed dogs, look at the condition of the people, stop the load shedding, by God we will burn you alive	Threatening	Individual
02	دونوں کی آنکھیں نکال دوں گی	I will take out the eyes of both	Threatening	Group
03	ابھی کچھ کچھ امید زندہ ہورہی ہے	There is some hope now	Non-threatening	NA
04	تم کتے کی وہ دم ہو جو بارہ سال بند رکھی پھر بھی ٹیڑی یہ کام تمہارہ مرشد ہی کرتا تھا	You are the dog's tail that was tied for twelve years, yet your mentor used to do this	Non-threatening	NA

solution. As described earlier, we are interested in fine-tuning of Urdu-BERT for threatening content and target identification tasks at tweet level. The reasons why we chose BERT model are provided below:

1. To capture the real context of the Urdu language used to describe the threatening materials while targeting some individual or group, BERT is the most robust method available as the language model. The prior approaches used frequency-based methods.
2. The fine-tuning of BERT enables us to retrain the language model to learn new knowledge about a specific problem while holding the previous learning.
3. The accuracy achieved by the prior approaches is not appreciable, fine-tuning BERT has already proved its significance in several NLP tasks. In addition, it demonstrated significant improvement in detecting threatening content and target type from Urdu tweets.

3.2.1. Fine-tunning

There are a few steps involved in fine-tuning BERT. We have to transform our input data into a specified format so that BERT can be trained on it. First, we pre-processed the tweet dataset to remove noise from it.

Pre-processing Steps: Hashtags, URLs, HTML tags, all punctuations, numbers, and white spaces are removed. Then normalization is performed.

Data Transformation: Then, each tweet is encountered by the tokenizer (uncased-BERT tokenizer) to split it into word tokens. This vocabulary of tokens is appended by the [SEP] token at the end and the [CLS] token at the start. The classification of the input data (tweets) is mainly dependent on the output embeddings generated from the [CLS] token. After that, each token is mapped to an index based on the pre-trained Urdu-BERT tokenizer vocabulary.

We are dealing with a tweet dataset and 280 characters are supported at maximum by Twitter for a single tweet. Therefore, the sequence length of 64 and 128 are enough for experimental setup and all the tweet dataset is then padded to a fixed length of 64 and 128. After that, attention masks are added so that padded and real tokens can be identified. The outcomes of attention masks are further used to fine-tune the BERT base classifier.

Training Classifier: The tweet dataset is divided into the train (80%) and test (20%) sets using a stratified sampling method, in which 20 percent of data is used for testing and 80 percent of data is used for training the BERT base classifier. Furthermore, during the process of fine-tuning, 10 % of training data is used as a validation set to validate the performance of BERT classifiers. We used the Urdu-BERT base model (Das et al., 2022), with 12 transformer layers, 768 hidden layers, and 12 attention heads. The maximum sequence length used in fine-tuning is 128. The training dataset is randomly split into 90–10 train-validation samples and the token ids, attention masks, and class labels of the dataset are combined into one set. For classification, we used a single layer added on top of the BERT base model. Regarding hyperparameters for fine-tuning, we took guidance from the researchers (Devlin et al., 2018). The batch sizes of 8, 16, and 32 are chosen for sequence lengths of 64 and 128 because we want to explore all possible configurations for 64 and 128 sequence lengths. Furthermore, the reason behind choosing only 64 and 128 sequence lengths is that the maximum number of tokens (maximum 280 characters allowed for a tweet) generated by a tweet is accommodated up to 128 sequence lengths. The optimizer function is used to update parameters for each epoch and the output of each training cycle of BERT is evaluated by calculating accuracy and validation loss on the validation dataset. Six classifiers of BERT are fine-tuned and the Google Colab platform is used to perform the operations. The hyperparameters used to fine-tune BERT are presented in Table 5.

Catastrophic Forgetting: Prior studies revealed that there is a risk of forgetting already learned knowledge while tuning weights for learning new knowledge. Scientists referred to it as catastrophic forgetting in transfer learning and the BERT model is prone to this effect (Sun et al., 2019). To examine the effects of catastrophic forgetting, we explored the following range of learning rates (1e-4, 1e-5, 2e-5, 3e-4,3e-5, 5e-5) for fine-tuning of BERT model. By unfreezing all layers of BERT in the fine-tuning process (Sun et al., 2019), we performed training repeatedly and monitored it carefully to finalize the starting learning rate. Our analysis concluded that with higher learning rates such as 3e-4, 3e-5, and 5e-5, fine-tuning of BERT leads to convergence failure. We got the best performance with a learning rate of 2e-5, to handle the problem of catastrophic forgetting.

Overfitting: In deep learning, choosing the appropriate number of epochs for training the model is a common issue. It is established that very few epochs may result in under-fitting whereas too many epochs lead to over-fitting. In our study, we use validation loss measure to monitor the performance of fine-tuning of BERT and train the model for 5 epochs. The validation loss was carefully monitored to see the performance improvement (discussed in Sections 4.1 And 4.3).

3.3. Latent semantic analysis (LSA)

LSA is a natural language processing technique to analyze the relationships between a set of documents and the keywords/terms contained in them to derive a set of concepts. LSA takes input in the form of bag-of-words and it aims to reduce the dimension of features for classification tasks (Dumais, 2004). It is based on the concept of words having similar meanings will appear in similar pieces of documents. In this study, the feature space is reduced to 100 dimensions and investigated their impact on threatening language and target identification tasks. We included the LSA model in our experimental setup to determine the effectiveness of fine-tuned BERT model by comparing it with LSA (another automated feature generation method).

3.4. Bag-of-words

The BOW model is one of the automated feature generation techniques and we used it in our study to compare with fine-tuned BERT model. The comparison will clarify the strength of fine-tuned BERT over the bag-of-words model.

3.5. Baselines

To compare the performance of the proposed framework with the state-of-the-art benchmark, we choose the study (Amjad et al., 2021b) as the baseline. This study utilized word n-gram (uni, bi, tri, and their combined effect), char n-gram (1-gram, 2-gram 3-gram, 4-gram, 5-gram, 6-gram, 7-gram, 8-gram, and their combined effect), and pre-trained FastText model with 300 dimensions.

Table 5
Hyperparamters used for fine-tuning of BERT model.

Sequence length	Epochs	Batch size	Learning Rate	Epsilon
64	1–5	8, 16, 32	2e-5	1e-8
128	1–5	8, 16, 32	2e-5	1e-8

3.6. Performance evaluations

We treat the threatening content and target identification as a binary classification problem. As described earlier, the dataset is split into train-test (80–20). The performance of all classifiers including fine-tuned BERT and baselines are evaluated using accuracy, precision, recall, and F1-score. For LSA, bag-of-words, and baselines, we compared Logistic Regression (LR), Random Forest (RF), SVM, K-nearest neighbor (KNN), and naïve Bayes (NB) classifiers and then describe the best one. The description of each classifier is provided below:

Logistic Regression: LR is a classification model rather than a regression model. It is a statistical model, designed for binary classification tasks but can be extended for multi-class classification (Subasi, 2020). It is more efficient for linearly separable class problems and utilizes the sigmoid function by transforming the linear regression method. We used the default parameters for model training and testing, Scikit-learn Python library is used.

Random Forest: RF is a popular machine learning model, commonly used for classification and regression problems. It combines the outputs of several decision trees to reach a single outcome. RF is the extension of the bagging ensemble model. It combines feature randomness and bagging to generate an uncorrelated forest of decision trees. The benefits include flexibility, reduction in the risk of overfitting, and feature importance can be determined easily (Liaw and Wiener, 2002).

Support Vector Machine: SVM is a supervised ML method commonly used for classification as well as regression tasks (Hearst et al., 1998). This algorithm aims to find the hyperplane of n-dimensional space that categorizes the data instances distinctly. The number of input features determines the dimensions of the hyperplane. The benefits of the SVM algorithm include: being effective when data is high-dimensional, different kernel functions are available, custom kernel functions can be specified, and it is memory efficient.

K-nearest neighbor: KNN is a supervised learning and non-parametric ML model, which uses the concept of neighborhood/proximity to do predictions for the neighbors of a data instance (Peterson, 2009). The final classification of the data instance is based on the majority voting of predicted labels about selected neighbors. This model can be used for classification or regression tasks, however, it is commonly used for classification.

Naive Bayes: The NB is another supervised ML algorithm mainly used for text classification (Rish, 2001). It is one of the generative learning algorithms that model the distributions of samples of a given class/category. The NB algorithm assumes a naïve property, which means features are conditionally independent. In addition, it assumes that all features contribute equally. The advantages of NB are: fast for training and prediction tasks, has a few parameters to tune, is easily interpretable, and facilitates probabilistic prediction.

4. Results and discussion

In this section, several experiments are conducted to fine-tune BERT classifiers for threatening content and target identification tasks. In addition, the impact of latent semantic analysis and bag-of-words models are explored including a state-of-the-art baseline to compare the effectiveness of BERT classifiers.

4.1. Fine-tunning of BERT (Threatening vs non-Threatening)

This section describes the results achieved by the six fine-tuned BERT models with sequence lengths of 64 and 128. The batch sizes are 8, 16, and 32 for 64 and 128 sequence lengths and the performance of BERT classifiers is presented in Table 6. The results include training and validation of BERT classifiers for two sequence lengths (64 and 128). Specifically, for each sequence length, batch size, training, and validation loss, validating accuracy, and training & validation times are reported. All experiments are performed

Table 6
Training and validation results for fine-tuning BERT (Threatening vs non-Threatening).

Sequence Length	Batch Size	Epochs	Training Loss	Validation Loss	Validation Accuracy	Training Time	Validation Time
BERT-64	8	1	0.65	0.60	0.783333	0:00:21	0:00:02
		2	0.36	0.56	0.841667	0:00:21	0:00:02
		3	0.22	0.55	0.858333	0:00:21	0:00:02
		4	0.13	0.63	0.854167	0:00:21	0:00:02
		5	0.08	0.69	0.866667	0:00:21	0:00:02
	16	1	0.71	0.66	0.750000	0:00:30	0.00.04
		2	0.39	0.61	0.800000	0:00:30	0.00.04
		3	0.23	0.61	0.820833	0:00:30	0.00.04
		4	0.15	0.70	0.829167	0:00:30	0.00.04
		5	0.10	0.75	0.816667	0:00:30	0.00.04
	32	1	0.78	0.74	0.695833	0:00:35	0:00:05
		2	0.49	0.72	0.745833	0:00:35	0:00:05
		3	0.34	0.71	0.812500	0:00:35	0:00:05
		4	0.25	0.77	0.829167	0:00:35	0:00:05
		5	0.18	0.80	0.833333	0:00:35	0:00:05
BERT-128	8	1	0.60	0.55	0.795833	0:00:40	0:00:08
		2	0.36	0.51	0.833333	0:00:40	0:00:08
		3	0.24	0.50	0.854167	0:00:40	0:00:08
		4	0.16	0.58	0.858333	0:00:40	0:00:08
		5	0.08	0.63	0.854167	0:00:40	0:00:08
	16	1	0.68	0.64	0.779167	0:00:43	0:00:10
		2	0.34	0.62	0.804167	0:00:43	0:00:10
		3	0.23	0.62	0.816667	0:00:43	0:00:10
		4	0.19	0.68	0.825000	0:00:43	0:00:10
		5	0.13	0.72	0.820833	0:00:43	0:00:10
	32	1	0.74	0.72	0.716667	0:00:45	0:00:12
		2	0.46	0.69	0.791667	0:00:45	0:00:12
		3	0.31	0.73	0.808333	0:00:45	0:00:12
		4	0.22	0.77	0.812500	0:00:45	0:00:12
		5	0.14	0.80	0.837500	0:00:45	0:00:12

using a 2e-5 learning rate, 1e-8 Epsilon, and 5 epochs for training, validation, and testing processes.

In the first step, the Urdu-BERT base model is trained and validated for the classification of threatening vs non-threatening content using a sequence length of 64 with 8, 16, and 32 batch sizes step by step. Here, training loss, validation loss, and validating accuracy are the important parameters to analyze the performance of BERT classifiers. For a batch size of 8, the training and validation loss are presented in Fig. 3**a**.

The best-validating accuracy is achieved on epoch 5 and it continuously increases up to epoch 5 whereas validation loss decreases up to epoch 3 but then starts increasing. This indicates that further training will result in overfitting. However, training loss continuously decreases to converge to a certain point. For a batch size of 16, the training and validation loss is presented in Fig. 3**b**. The training loss decreases from 0.70 to 0.10 from epoch 1 to epoch 5 whereas validation loss decreases from 0.66 to 0.61 from epochs 1–3 but then increases from 0.70 to 0.75 from epochs 4–5. The best-validating accuracy is achieved on epoch 4. For a batch size of 32, the accuracy increases from 69.58% to 83.33% (continuously increasing) whereas validation loss starts decreasing up to epoch 3 (0.74 to 0.71) but then starts increasing up to epoch 5 as shown in Fig. 3**c**.

Likewise, for the sequence length of 128, we trained and validated BERT classifiers for 8, 16, and 32 batch sizes as shown in the lower part of Table 6. The validating accuracy increases from 79.58% to 85.83% from epochs 1–4. However, validating loss decreases from epochs 1–3 and then increases steadily up to epoch 5, revealing that further training will lead to overfitting. In contrast, training loss steadily decreases from epochs 1–5 when the

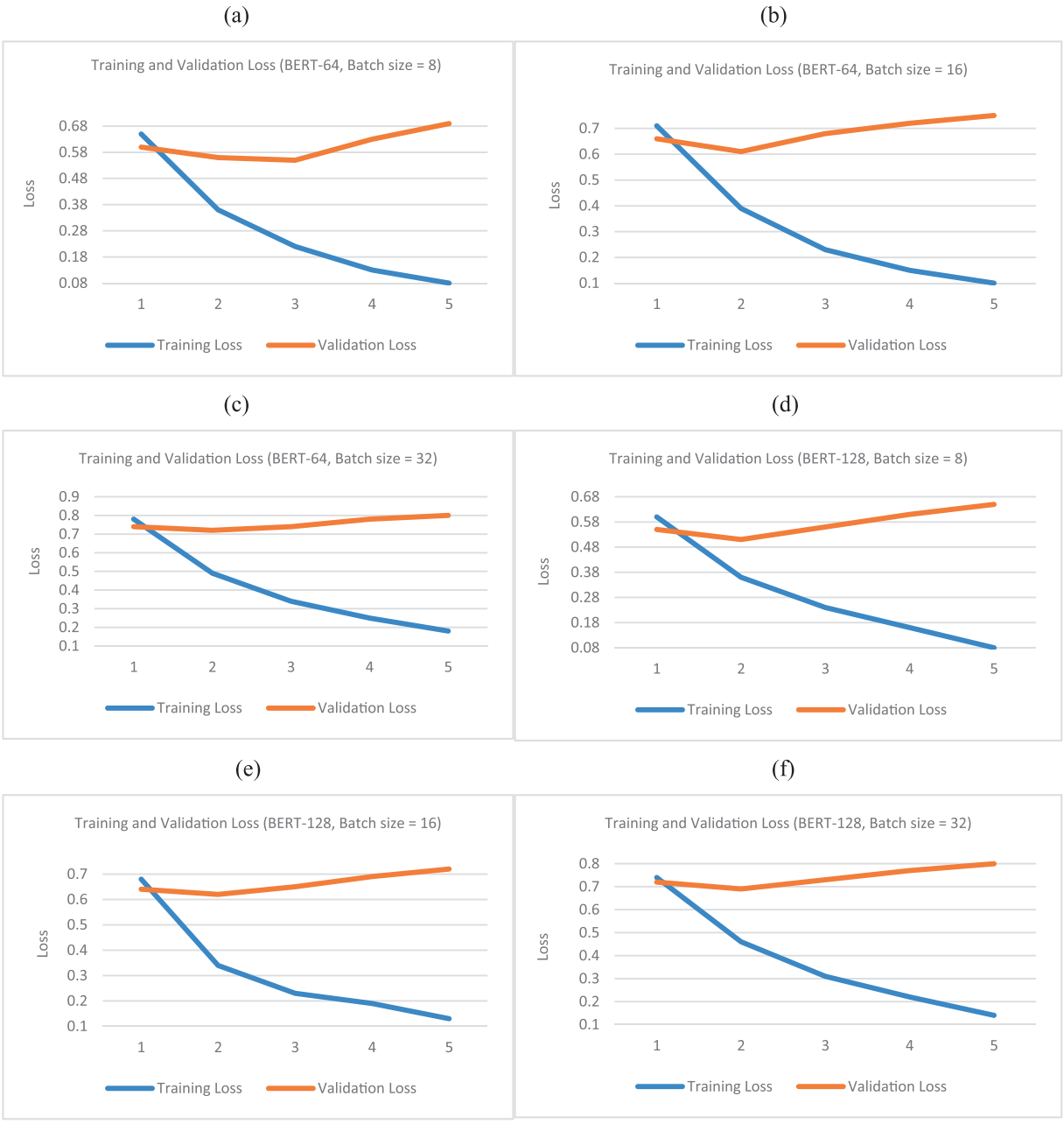

Fig. 3. Training and Validation loss for sequence length of 64 with a) batch-size 8, b) batch-size 16, c) batch-size 32 and sequence length of 128 with d) batch-size 8, e) batch-size 16, f) batch-size 32.

model is trained for 8, 16, and 32 batch sizes. We observed a symmetrical behavior for validating loss as it decreases from epochs 1–3 and then starts increasing as shown in Fig. 3**(d)(e)(f)**.

By comparing the performance of BERT classifiers, we can conclude that validation loss has almost symmetrical behavior by decreasing for the first three epochs and then increasing up to the fifth epoch. In contrast, training loss decreases steadily and exhibits symmetrical behavior for five epochs, however, validating accuracy increases continuously for the first four epochs but demonstrates random behavior on the fifth epoch. The best accuracy of 86.67% is obtained with a sequence length of 64 and batch size of 8, in contrast, the worst accuracy is obtained with a sequence length of 64 and batch size of 32.

4.2. Comparison of Fine-tuned BERT with others (Threatening vs non-Threatening)

In this section, the comparison of fine-tuned BERT classifiers is carried out with LSA, BOW, and and state-of-the-art baseline (Amjad et al., 2021b). For LSA, bag-of-words, and baseline features, we used LR, RF, SVM, KNN, and NB classifiers, and results are demonstrated in Fig. 4. It is observed that not a single classifier showed the best performance for all type of features, however, LR outperformed the others for the majority of feature types. Therefore, we have chosen the results of LR for comparison with fine-tuned BERT classifiers.

Table 7 shows the performance of all feature models including fine-tuned BERT and baseline (Amjad et al., 2021b). The metrics are accuracy, precision, recall and F1-score computed using the confusion matrix generated by the test part of the dataset. Considering the baseline, Char 5-gram showed better performance than other features of the baseline with an accuracy of 86.25% and an F1-score of 85.83%. The performance of the char 6-gram is comparable with the char 5-gram model. In comparison with the baseline, LSA and bag-of-word models demonstrated comparable performance on all evaluation metrics. For BERT classifiers, we added one entry (best one) against each sequence length and batch size as shown in Table 7. The best performance of the BERT classifier is obtained with a sequence length of 64 and batch size of 8, leading to 87.5% accuracy and 87.8% F1-score. Moreover, the BERT classifier demonstrated higher metric values with small batch sizes as compared to large batch sizes. With a sequence length of 128, the best performance is observed against a batch size of 8. However, the lowest accuracy and F1-score are observed for a sequence length of 128 and batch size of 32 considering only BERT classifiers. For the baseline, FastText presented the lowest performance in accuracy and F1-score. Thus the fine-tuning of BERT improves 1.2% accuracy and 2% F1-score as compared to the state-of-the-art baseline for threatening vs non-threatening tweet identification.

4.3. Fine-tunning of BERT and comparison with others (Target Identification)

In this section, we investigate the impact of fine-tuning BERT classifiers for target identification (individual vs group) in comparison with LSA, BOW, and state-of-the-art baseline (Amjad et al., 2021b). Here, again six classifiers of the BERT model are trained, validated, and tested with sequence lengths of 64 and 128 and batch sizes of 8, 16, and 32. The training and validation of six BERT models are presented in Table 8. The same parameters and configurations are used to train and validate the BERT classifiers as chosen in the prior experiment (Table 6).

Here the objective is target identification (individual vs group) from tweet text. The training and validation loss and validation accuracy are calculated for five epochs. For the sequence length of 64, training loss decreases steadily after each epoch and converges to a certain threshold for all batch size configurations (8, 16, and 32). In contrast, the validation loss starts decreasing up to the 2nd epoch and then starts increasing up to the 5th epoch for 8, 16, and 32 batch sizes as shown in Fig. 5. This indicates that further training may lead to overfitting. The response of validation accuracy is slightly different; For a batch size of 8 (with a sequence length of 64), it continuously increases and achieves a 79.16% threshold whereas, for a batch size of 16, it increases up to epoch 3, decreases for epoch 4, and then increases for epoch 5. For a batch size of 32, accuracy increases up to epoch 4 and then decreases on epoch 5. The best-validation accuracy is obtained at epoch 5 (79.16%), with a batch size of 8 while observing a sequence length of 64.

Likewise, the performance of BERT classifiers on a sequence length of 128 is presented in the lower part of Table 8. Three batch sizes (8, 16, 32) are used to train and validate the classifiers. The response of training loss is similar (continuously decreasing) to the training loss obtained by a sequence length of 64. In contrast, validation loss decreases up to epoch 2 and then starts increasing

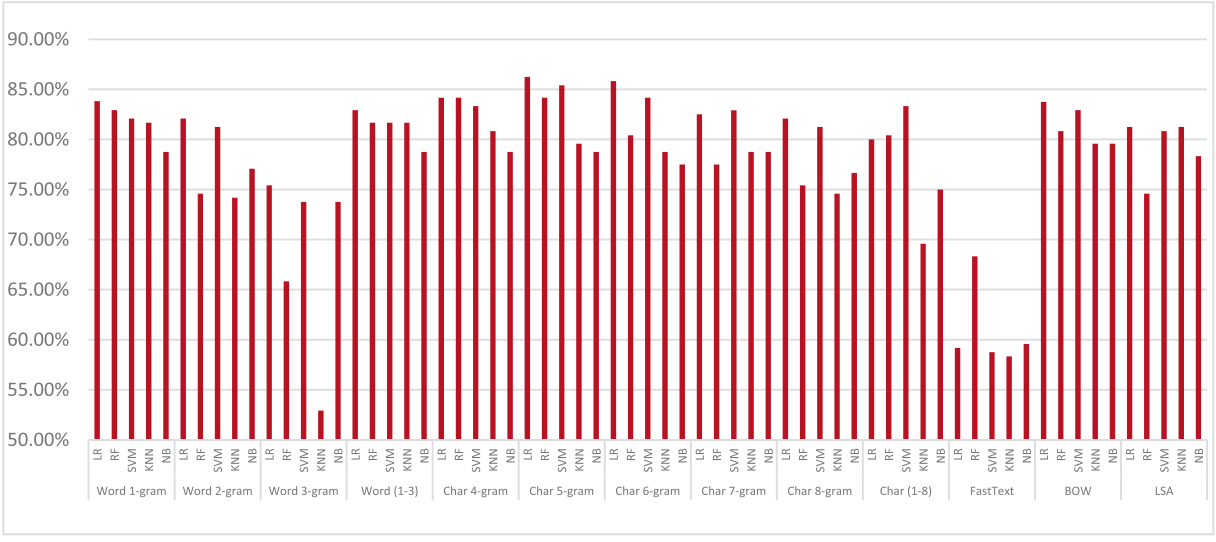

Fig. 4. Performance of logistic regression, random forest, support vector machine, K-nearest neighbor, and naïve Bayes classifiers on LSA, BOW, and baseline features (accuracy based).

Table 7
Comparison of fine-tuned BERT and baselines (Threatening vs non-Threatening).

Type	Features	Accuracy	Precision	Recall	F1-score
Baseline (Amjad et al., 2021b)	Word uni-gram	83.83	81.51	80.17	80.83
	Word bi-gram	82.08	81.97	82.64	82.30
	Word tri-gram	75.42	80.39	67.77	73.54
	Word combined (1-2-3)	82.92	80.30	87.60	83.79
	Char 1-gram	68.75	69.16	68.59	68.87
	Char 2-gram	79.58	82.14	76.03	78.97
	Char-3-gram	80.00	84.11	74.38	78.94
	Char 4-gram	84.16	87.38	80.16	83.62
	Char 5-gram	**86.25**	**89.28**	**82.64**	**85.83**
	Char 6-gram	85.83	87.82	83.47	85.59
	Char 7-gram	82.50	83.19	81.81	82.50
	Char 8-gram	82.08	82.50	81.81	82.15
	Char combined (1–8)	80.00	74.82	90.90	82.08
	FastText	59.17	63.64	54.69	58.82
Proposed	Bag of words	83.75	81.54	87.60	84.46
	Latent Semantic Analysis	81.25	79.23	85.12	82.07
	BERT-64 (8)	**87.5**	**86.4**	**89.26**	**87.8**
	BERT-64 (16)	83.75	85.34	81.82	83.54
	BERT-64 (32)	84.58	87.5	80.99	84.12
	BERT-128 (8)	**87.5**	**89.57**	**85.12**	**87.29**
	BERT-128 (16)	84.17	86.73	80.99	83.76
	BERT-128 (32)	82.92	80.3	87.6	83.79

Table 8
Training and validation results for fine-tuning BERT (Individual vs Group).

Sequence Length	Batch Size	Epochs	Training Loss	Validation Loss	Validity Accuracy	Training Time	Validation Time
BERT-64	8	1	0.73	0.72	0.691667	0:00:19	0:00:02
		2	0.46	0.70	0.766667	0:00:19	0:00:02
		3	0.32	0.74	0.775000	0:00:19	0:00:02
		4	0.22	0.78	0.783333	0:00:19	0:00:02
		5	0.14	0.83	0.791667	0:00:19	0:00:02
	16	1	0.71	0.69	0.658333	0:00:28	0:00:03
		2	0.49	0.67	0.741667	0:00:28	0:00:03
		3	0.35	0.70	0.741667	0:00:28	0:00:03
		4	0.29	0.74	0.658333	0:00:28	0:00:03
		5	0.19	0.79	0.750000	0:00:28	0:00:03
	32	1	0.77	0.75	0.608333	0:00:33	0:00:05
		2	0.58	0.73	0.675000	0:00:33	0:00:05
		3	0.46	0.76	0.708333	0:00:33	0:00:05
		4	0.35	0.80	0.750000	0:00:33	0:00:05
		5	0.25	0.83	0.716667	0:00:33	0:00:05
BERT-128	8	1	0.70	0.68	0.650000	0:00:39	0:00:07
		2	0.44	0.67	0.700000	0:00:39	0:00:07
		3	0.31	0.69	0.758333	0:00:39	0:00:07
		4	0.20	0.73	0.775000	0:00:39	0:00:07
		5	0.10	0.76	0.791667	0:00:39	0:00:07
	16	1	0.70	0.67	0.675000	0:00:42	0:00:10
		2	0.51	0.65	0.700000	0:00:42	0:00:10
		3	0.36	0.68	0.791667	0:00:42	0:00:10
		4	0.23	0.72	0.783333	0:00:42	0:00:10
		5	0.17	0.75	0.766667	0:00:42	0:00:10
	32	1	0.80	0.76	0.608333	0:00:46	0:00:12
		2	0.59	0.72	0.708333	0:00:46	0:00:12
		3	0.47	0.75	0.691667	0:00:46	0:00:12
		4	0.38	0.78	0.758333	0:00:46	0:00:12
		5	0.30	0.82	0.750000	0:00:46	0:00:12

up to epoch 5 for all batch sizes as shown in Fig. 5, indicating that further training will lead to overfitting. However, the response of validation accuracy is different; it is steadily increasing for a batch size of 8 whereas it starts increasing for the first three epochs and then decreases for a batch size of 16. For a batch size of 32, it is random. The best accuracy is 79.16% and it was obtained by a batch size of 8 and 16. The worst accuracy is observed with 32 batch size on epoch 1.

Next, we are interested to compare the performance of BERT classifiers with state-of-the-art baseline (Amjad et al., 2021b), LSA, and BOW models. The results are presented in Table 9 and four evaluation metrics (accuracy, precision, recall, and F1-score) are used to compare the performances. As described earlier that the baseline dataset has issues and the maximum performance achieved for the target identification task was 75.31% accuracy and 57.84% F1-score with the SVM classifier using FastText pre-trained model. In contrast, we designed a new annotated dataset for threatening content and target identification task. We recalculated the results of the baseline on our dataset. For the baseline, the best results are presented by the word combined (1-2-3 g) model by demonstrating 82.92% accuracy, and 83.79% F1-score. However, for the threatening content identification task, char 5-gram

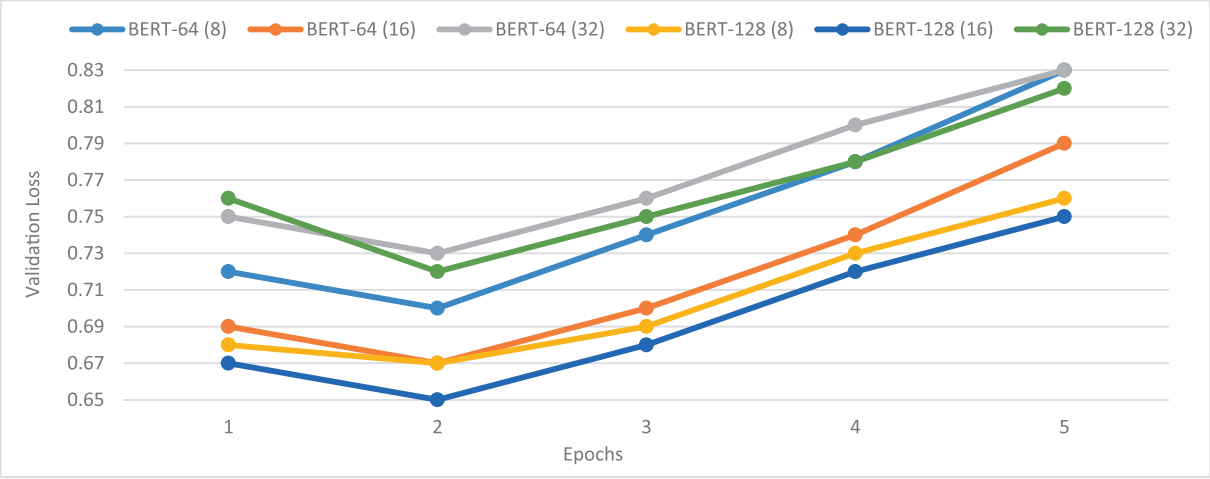

Fig. 5. Validation loss generated by BERT classifiers for sequence lengths of 64 and 128 with 8, 16, and 32 batch sizes (Target Identification).

Table 9
Comparison of fine-tuned BERT with baselines (Target Identification).

Type	Feature Set	Accuracy	Precision	Recall	F1-score
Baseline (Amjad et al., 2021b)	Word uni-gram	80.33	81.51	80.17	80.83
	Word bi-gram	82.08	81.97	82.64	82.30
	Word tri-gram	75.42	80.39	67.77	73.54
	Word combined (1-2-3)	**82.92**	**80.30**	**87.60**	**83.79**
	Char 1-gram	60.83	68.08	50.00	57.65
	Char 2-gram	69.16	72.13	68.75	70.40
	Char 3-gram	70.83	72.30	73.43	72.86
	Char 4-gram	70.83	73.77	70.31	72.00
	Char 5-gram	69.16	71.42	70.31	70.86
	Char 6-gram	67.50	71.18	65.62	68.29
	Char 7-gram	60.83	66.66	53.12	59.13
	Char 8-gram	61.66	68.75	51.56	58.92
	Char combined (1–8)	72.50	79.24	65.62	71.79
	FastText	54.17	51.67	54.39	52.99
Proposed	Bag-of-words	**80.33**	**81.51**	**80.17**	**80.83**
	Latent Semantic Analysis	68.33	68.57	75.00	71.64
	BERT-64 (8)	**79.17**	**80**	**81.25**	**80.62**
	BERT-64 (16)	75	81.48	68.75	74.58
	BERT-64 (32)	75	79.31	71.88	75.41
	BERT-128 (8)	**82.5**	**85.25**	**81.25**	**83.2**
	BERT-128 (16)	79.17	81.97	78.13	80
	BERT-128 (32)	76.67	83.33	70.31	76.27

presented the best results in contrast to the word combined (1-2-3 g) model. Furthermore, word n-gram (uni, bi, etc) models demonstrated better performances than char n-grams and the FastText model. The worst performance is obtained by the FastText model.

For the proposed models, performances of six BERT classifiers are added in the lower part of Table 9 in addition to BOW and LSA models. BERT-128 with a batch size of 8 presented the best results as compared to other BERT classifiers, LSA, and BOW models by obtaining 82.5% accuracy, and 83.2% F1-score. This performance is comparable with the word combined (1-2-3 g) model. The second best-performing proposed model is the BOW model. The BERT-64 with a batch size of 8 demonstrates the third-best performance. The LSA model shows the lowest performance for the target identification task. This completes the analysis of the results. Thus, our automated feature generation method (Fine-tuning of BERT) showed better than the baseline for the threatening content identification task. Furthermore, it (fine-tuned BERT) showed comparable performance with the word combined (1-2-3 g) model for the target identification task.

At last, we summarize that the advantages of our proposed method are three-fold. First, BERT is an automated feature-generation model in contrast to feature models (hand-crafted features) used in the baseline. Second, prior approaches are based on frequency-based features, but language models like BERT can capture the actual context of the language being used to threat and target someone. Third, the proposed framework improved the accuracy of threatening content identification as compared to the benchmark study.

5. Conclusion

This research investigated the problem of threatening content and target identification at the tweet level in the Urdu language. A new data corpus is constructed for the experimental setup. The dataset is balanced and consists of annotation for threatening vs non-threatening at the first level and individual vs group at the second level. To support the generalization of the solution, fine-tuning of BERT is explored with two sequence lengths and three batch sizes without relying on hand-crafted features. In addition, LSA and BOW models are explored. Six classifiers for BERT are trained, evaluated, and tested in comparison with the baseline. The results showed that BERT-64 with a batch size of 8 outperformed the baseline and all other models by improving 1.2% accu-

racy and 2% F1-score for threatening content identification. This model achieved 87.5% accuracy, 86.4% precision, 89.26% recall, and 87.8% F1-score. Furthermore, BERT-128 with a batch size of 8 presented 87.5% accuracy, 89.57% precision, 85.12% recall, and 87.29% F1-score. For the target identification task, BERT-128 with a batch size of 8 attained 82.5% accuracy, 85.25% precision, 81.25% recall, and 83.2% F1-score and demonstrated comparable performance with the baseline. The findings of current research help law and enforcement organizations to identify threatening content and target identification at the tweet level in the Urdu language.

In the future, this work can be extended by annotating a bag corpus in the Urdu language so that results can be generalized. The state-of-the-art variant of language models such as Roberta, XLNet, ALBERT, and Distil BERT could be utilized to capture the context effectively, this will lead to an improved classification system. The multilingual framework could be another extension to utilize transfer learning techniques for cross-language tasks.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This article is an output of a research project implemented as part of the Basic Research Program at the National Research University Higher School of Economics (HSE University). Moreover, this research was supported in part by computational resources of HPC facilities at HSE University.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

Akhter, M.P., Jiangbin, Z., Naqvi, I.R., Abdelmajeed, M., Zia, T., 2021. Abusive language detection from social media comments using conventional machine learning and deep learning approaches. Multimedia Syst., 1–16

Amjad, M., Ashraf, N., Sidorov, G., Zhila, A., Chanona-Hernandez, L., Gelbukh, A., 2021a. Automatic abusive language detection in urdu tweets. Acta Polytechnica Hungarica, 1785–8860.

Amjad, M., Ashraf, N., Zhila, A., Sidorov, G., Zubiaga, A., Gelbukh, A., 2021b. Threatening language detection and target identification in Urdu tweets. IEEE Access 9, 128302–128313.

Ashraf, N., Mustafa, R., Sidorov, G., Gelbukh, A., 2020,. Individual vs. group violent threats classification in online discussions. In: Companion Proceedings of the Web Conference 2020, pp. 629–633.

Balakrishnan, V., Khan, S., Arabnia, H.R., 2020. Improving cyberbullying detection using Twitter users' psychological features and machine learning. Comput. Secur. 90, 101710.

Behzadan, V., Aguirre, C., Bose, A., Hsu, W. Corpus and deep learning classifier for collection of cyber threat indicators in twitter stream. In: 2018 IEEE International Conference on Big Data (Big Data), IEEE, 2018, pp. 5002–5007.

Beyhan, F., Çarik, B., Arin, İ., Terzioğlu, A., Yanikoglu, B., Yeniterzi, R.A., 2022. Turkish hate speech dataset and detection system. In: Proceedings of the Thirteenth Language Resources and Evaluation Conference, pp. 4177–4185.

Das, M., Banerjee, S., Mukherjee, A., 2022. Data bootstrapping approaches to improve low resource abusive language detection for indic languages. In: Proceedings of the 33rd ACM Conference on Hypertext and Social Media, pp. 32–42.

Das, M., Banerjee, S., Saha, P., 2021. Abusive and threatening language detection in urdu using boosting based and bert based models: A comparative approach. *arXiv preprint arXiv:2111.14830*.

Del Vigna12, F., Cimino23, A., Dell'orletta, F., Petrocchi, M., Tesconi, M., 2017. Hate me, hate me not: Hate speech detection on facebook. In: Proceedings of the First Italian Conference on Cybersecurity (ITASEC17), pp. 86–95.

Devlin, J., Chang, M.-W., Lee, K., Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dumais, S.T., 2004. Latent semantic analysis. Annu. Rev. Inf. Sci. Technol. 38, 188–230.

Febriana, T., Budiarto, A., 2019. Twitter dataset for hate speech and cyberbullying detection in Indonesian language. In: 2019 International Conference on Information Management and Technology (ICIMTech), IEEE, pp. 379–382.

Fleiss, J.L., 1971. Measuring nominal scale agreement among many raters. Psychol. Bull. 76, 378.

Hajibabaee, P., Malekzadeh, M., Ahmadi, M., Heidari, M., Esmaeilzadeh, A., Abdolazimi, R., James, J.R.H., 2022. Offensive language detection on social media based on text classification. In: 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC), IEEE, pp. 0092–0098.

Hearst, M.A., Dumais, S.T., Osuna, E., Platt, J., Scholkopf, B., 1998. Support vector machines. IEEE Intell. Syst. Appl. 13, 18–28.

Humayoun, M., 2022. Abusive and Threatening Language Detection in Urdu using Supervised Machine Learning and Feature Combinations. *arXiv preprint arXiv:2204.03062*.

Hussain, S., Malik, M.S.I., Masood, N., 2022. Identification of offensive language in Urdu using semantic and embedding models. PeerJ Comput. Sci. 8, e1169.

Jiang, L., Shi, J., Wang, C., Pan, Z., 2023. Intelligent control of building fire protection system using digital twins and semantic web technologies. Autom. Constr. 147, 104728.

Kalraa, S., Agrawala, M., Sharmaa, Y., 2021. Detection of Threat Records by Analyzing the Tweets in Urdu Language Exploring Deep Learning Transformer-Based Models.

Kamal, A., Anwar, T., Sejwal, V.K., Fazil, M., 2023. BiCapsHate: attention to the linguistic context of hate via bidirectional capsules and hatebase. IEEE Trans. Comput. Social Syst.

Liaw, A., Wiener, M., 2002. Classification and regression by randomForest. R news 2, 18–22.

Malik, M.S.I., Imran, T., Mamdouh, J.M., 2023. How to detect propaganda from social media? Exploitation of semantic and fine-tuned language models. PeerJ Comput. Sci. 9, e1248.

Mazari, A.C., Boudoukhani, N., Djeffal, A., 2023. BERT-based ensemble learning for multi-aspect hate speech detection. Clust. Comput., 1–15

Mehmood, A., Farooq, M.S., Naseem, A., Rustam, F., Villar, M.G., Rodríguez, C.L., Ashraf, I., 2022. Threatening URDU language detection from tweets using machine learning. Appl. Sci. 12, 10342.

Mubarak, H., Hassan, S., Chowdhury, S.A., 2022. Emojis as anchors to detect arabic offensive language and hate speech. *arXiv preprint arXiv:2201.06723*.

Nelatoori, K.B., Kommanti, H.B., 2022. Attention-based bi-lstm network for abusive language detection. IETE J. Res., 1–9

Peterson, L.E., 2009. K-nearest neighbor. Scholarpedia 4, 1883.

Rani, P., Ojha, A.K., 2019. KMI-coling at SemEval-2019 task 6: exploring N-grams for offensive language detection. In: Proceedings of the 13th International Workshop on Semantic Evaluation, pp. 668–671.

Razavi, A.H., Inkpen, D., Uritsky, S., Matwin, S., 2010. Offensive language detection using multi-level classification. In: Advances in Artificial Intelligence: 23rd Canadian Conference on Artificial Intelligence, Canadian AI 2010, Ottawa, Canada, May 31–June 2, 2010. Proceedings 23, 2010. Springer, pp. 16–27.

Rish, I., 2001. An empirical study of the naive Bayes classifier. In: IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence, pp. 41–46.

Saeed, R., Afzal, H., Rauf, S.A., Iltaf, N., 2023. Detection of offensive language and its severity for low resource language. ACM Trans. Asian Low-Resource Language Informat. Process.

Saleh, H., Alhothali, A., Moria, K., 2021. Detection of Hate Speech using BERT and Hate Speech Word Embedding with Deep Model. *arXiv preprint arXiv:2111.01515*.

Schmidt, A., Wiegand, M., 2017. A survey on hate speech detection using natural language processing. In: Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media, pp. 1–10.

Shang, Y., 2017. Subgraph robustness of complex networks under attacks. IEEE Trans. Syst. Man Cybernet.: Syst. 49, 821–832.

Sigurbergsson, G.I., Derczynski, L., 2019. Offensive language and hate speech detection for Danish. *arXiv preprint arXiv:1908.04531*.

STATISTA *Statista. Number of Social Media Users Worldwide from 2018 to 2027*.

Subasi, A., 2020. Practical Machine Learning for Data Analysis Using Python. Academic Press.

Sun, C., Qiu, X., Xu, Y., Huang, X., 2019. How to fine-tune bert for text classification? In: Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18, Springer, pp. 194–206.

Wadud, M.A.H., Kabir, M.M., Mridha, M., Ali, M.A., Hamid, M.A., Monowar, M.M., 2022. How can we manage offensive text in social media-a text classification approach using LSTM-BOOST. Int. J. Informat. Manage. Data Insights 2, 100095.

Wang, X., Liu, Y., Sun, C.-J., Wang, B., Wang, X., 2015. Predicting polarities of tweets by composing word embeddings with long short-term memory. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 1343–1353.